AD NUMBER

ADB015924

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; OCT 1976. Other requests shall be referred to Rome Air Development Center, Griffiss AFB, NY 13441.

AUTHORITY

RADC ltr 9 Apr 1981

RADC-TR-76-318, Volume I (of four)
Final Technical Report
October 1976

ASSOCIATIVE PROCESSOR APPLICATION STUDY
Summary

Boeing Computer Services Inc

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

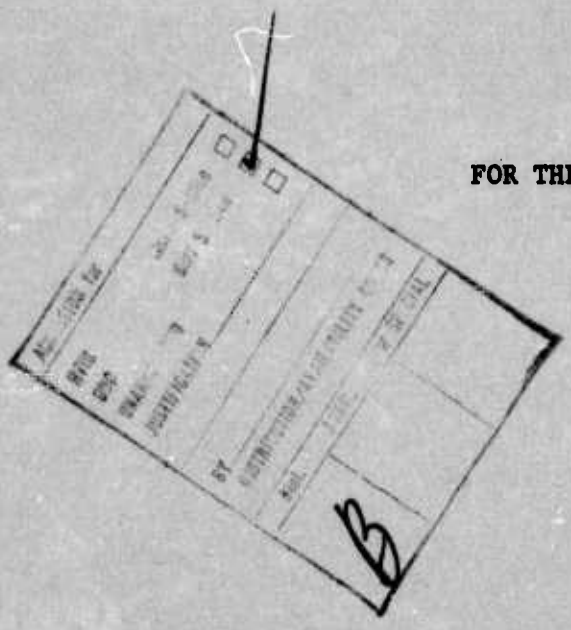
This report has been reviewed and approved for publication.

APPROVED: Alan R. Klayton

ALAN R. KLAYTON, Capt, USAF
Project Engineer

APPROVED: Robert D. Krutz

ROBERT D. KRUTZ, Col, USAF
Chief, Information Sciences Division



FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: RADC-TR-76-318-Vol-1 (of four)	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)*: ASSOCIATIVE PROCESSOR APPLICATION STUDY - Volume I Summary	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. Jan 75 - May 76
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): Hsiung-Fei Lee, Robert Katz, Thomas Fisher, Frederick Schenfele, Gordon Thomas, Ray Komada	8. CONTRACT OR GRANT NUMBER(s): F30602-75-C-0112
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Boeing Computer Services Inc, P. O. Box 24346, Seattle WA 98124	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 63728F, 55500127
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (ISCA), Griffiss AFB NY 13441	12. REPORT DATE: October 1976
	13. NUMBER OF PAGES: 28
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*: Same	15. SECURITY CLASS. *(of this report)*: UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT *(of this Report)*

Distribution limited to U. S. Gov't agencies only; test and evaluation; October 1976. Other requests for this document must be referred to RADC (ISCA), Griffiss AFB NY 13441.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer:
Capt Alan R. Klayton (ISCA)

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Associative Processors, Parallel Processors, Performance Evaluation, AWACS Surveillance Routines, Passive Tracking Function, Coordinate Conversion Function, Radar Data Correlation Function

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report describes work performed in support of an associative processor application study. A GAC STARAN-1000 associative processor (AP) was used to evaluate the capability of a parallel computer architecture to satisfy real time, command and control data processing requirements of the USAF. The passive tracking, radar data correlator and coordinate conversion functions of the AWACS were selected for use in the study. The passive tracking function was designed and implemented on the AP. The performance of the parallel program was compared against that of the sequential version which was measured on an

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

LB

390340

IBM 360/65 computer. The parallel passive tracking program used floating point arithmetic. The coordinate conversion function was also designed and implemented in fixed point arithmetic on the AP. The performance of the parallel program was measured. The radar data correlator function design for AP implementation used a new concept in radar signal processing. No performance data was collected.

TABLE OF CONTENTS

LIST OF FIGURES

EVALUATION

The work performed under this contract has clearly demonstrated the advantages realizable through the utilization of associative (parallel) processing techniques for satisfying high-data-rate real time command and control Air Force processing requirements. The effort represents the successful completion of one aspect of RADC's comprehensive investigation into associative processing for Air Force applications (TPO-12, FY-75, 76).

The results of this effort broadens the available knowledge base against which the processing systems for future AF command and control requirements can be evaluated. It also provides a foundation for the investigation of other concurrent and distributed processing systems which are now receiving much increased attention as a result of great strides in the development of Large Scale Integration techniques.

Alan R Klayton

ALAN R. KLAYTON, Capt, USAF
Project Engineer

1.1 INTRODUCTION

This final report describes the work accomplished by Boeing Computer Services, Inc. (BCS) under contract F30602-75-C-0112, "Associative Processor Application Study" for Rome Air Development Center (RADC). The objective of this effort was to study the applicability of using associative processing to meet the real-time, high data rate processing requirements of the USAF. The RADC testbed used in the study consists of a GAC STARAN-1000 Associative Processor (AP) interfaced with a HIS-6180 computer running under MULTICS operating system. The work started in January of 1975 and concluded in March of 1976.

Several AWACS functions which seemed to be suitable for parallel processing were analyzed and selected for RADCAP testbed implementation. These functions were passive tracking, coordinate conversion and radar data correlator. Both the passive tracking function and the coordinate conversion function were implemented on the GAC STARAN-1000. The performancesof the parallel processing programs were measured on the AP and analyzed. The performance of the sequential version of the passive tracking program was measured on an IBM 360/65 computer and the results were compared against those of the parallel version. The requirements and design of the radar data correlator AP implementation were completed.

1.2 SUMMARY

1.2.1 Passive Tracking Function

1.2.1.1 Program Descriptions (Parallel and Sequential)

One of the primary functions of the AWACS surveillance system is to provide tracking data on airborne targets. Tracking can be divided into two categories. These are active tracking and passive tracking. Active tracking is accomplished by using radar sensors which measure the range, range rate and azimuth of targets. Passive tracking is accomplished by using target azimuth information only. The target locations are determined from passive data by using triangulation. Two methods of passive tracking are employed. The first method, known as self-passive tracking, requires the AWACS to fly in a closed loop. When a second AWACS is available, its radar measurements can be cross-told to those of the primary AWACS. This method is known as cooperative passive tracking.

Passive tracking is primarily used to locate and track jamming targets from measurements of the azimuth of the radiation source. The radar returns which are referred to as strobes provide azimuth and strobe width data. A target normally operates its jamming equipment intermittently. When the jammer is off, active radar reports from the target may be obtained. When the passive tracking algorithms take advantage of this additional information, it is called active tracking.

The passive tracking algorithms perform four major functions. Each of these functions is performed each radar scan. The position and speed of each target is predicted for the current scan using the previous scan information. After the radar data has been read for a particular scan, each observed strobe is associated and correlated to a particular track. These values are then smoothed using the past track history. If cooperative passive tracking is employed, ghosts produced by multiple intersections of radar strobes are identified. Figure 1 shows the functional relationship.

The passive tracking program will accept as input, simulated radar data describing targets and the position of each AWACS, every radar scan. From this input, the program will identify individual tracks and will compute the position and speed of each target every radar scan. Both self-passive tracking and cooperative passive tracking will be performed. The program will initiate tracks by using an algorithm which simulates the AWACS operator function. Both passive radar reports (azimuth and strobe width) and active radar reports (azimuth and range) will be accepted by the program.

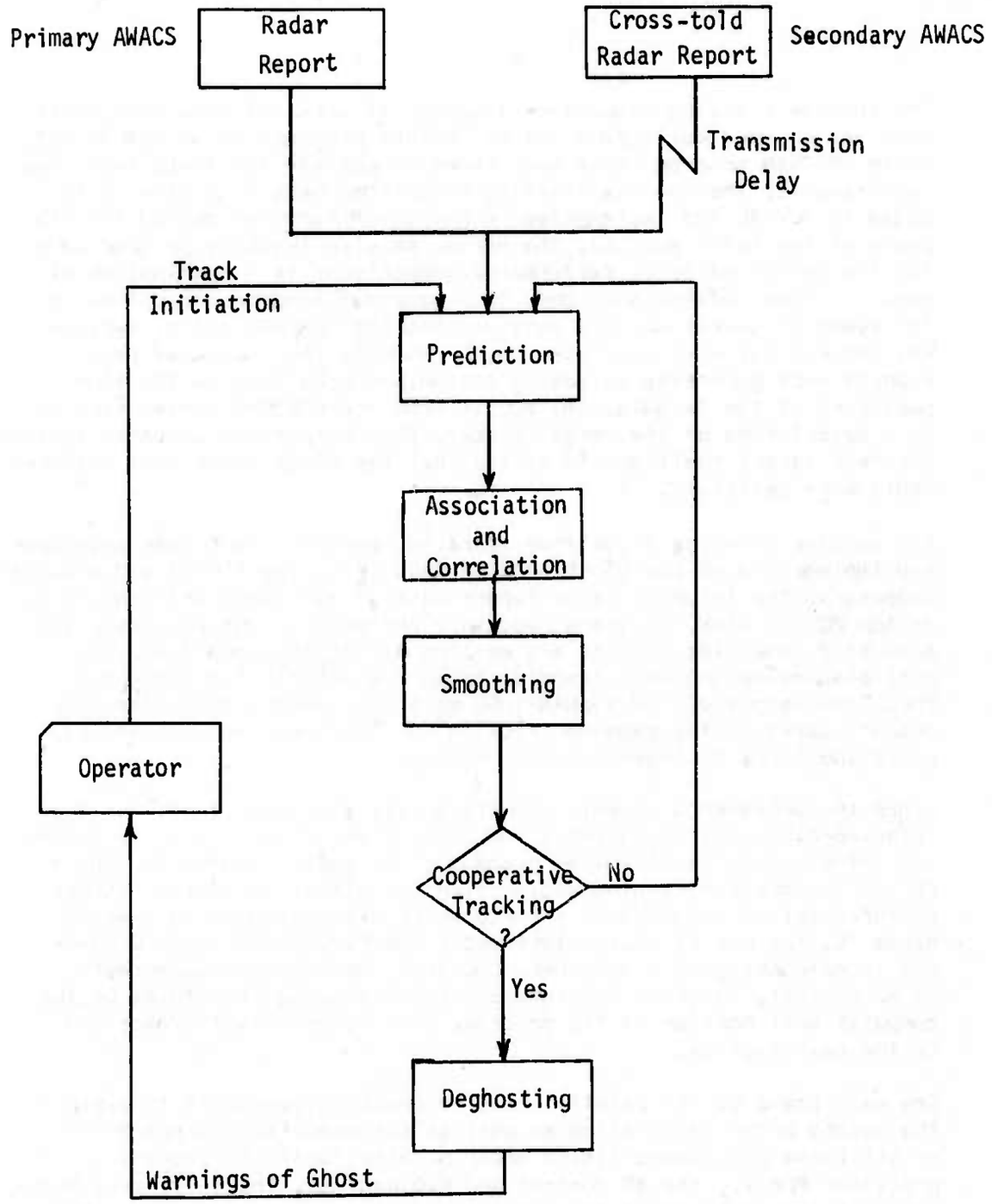


Figure 1 Passive Tracking Functional Relationships

The passive tracking algorithms (sequential version) have been developed and checked out with a set of FORTRAN programs on an IBM 360/65. These FORTRAN programs have been shown to satisfy the AWACS functional requirements. The passive tracking algorithms were also previously coded in JOVIAL and implemented on an IBM-4π computer during the SID phase of the AWACS project. The serial passive tracking program used for the serial/parallel performance comparisons is a combination of some of these FORTRAN programs. This combined program was optimized for speed of execution. The passive tracking FORTRAN serial version was checked out with simulated radar reports. The simulated radar reports were generated by adding random perturbations to the time positions of the targets. The output from the FORTRAN serial version is a description of the target tracks. This output was compared against the true target positions to verify that the AWACS functional requirements were satisfied.

The passive tracking algorithms (parallel version) have been developed and implemented on the GAC STARAN at RADC using the PDP-11 and HIS-6180 computers. The incoming radar report data of all scans are pre-stored on the PDP-11 disk. They are read into the program and processed one scan at a time. The results are written out on the same disk. The data processing and data transfer tasks are distributed among all three processors of the STARAN-1000 to allow operation overlap. The overall speed of the passive tracking functions can be improved by utilizing three processors simultaneously.

Since the processors operate simultaneously and their functions are inter-related, the synchronization among them is important. AP control initiates and halts all other processes. AP control waits for the PDP-11 to complete reading and writing (to disk). AP control issues synchronization commands to the PIO which are analogous to shaking hands (by the use of four interlocks). The PDP-11 and the PIO operate independently of each other. However, due to their dependence on AP control, indirect influences occur affecting the timing of the computational portion of the program. This is more fully described in the next section.

The main theme of the parallel passive tracking design is to explore the second order parallelism as well as the simultaneous operation of all three processors (third order parallelism): the sequential processor PDP-11, the AP control and PIO control. There are many mathematical computations in the passive tracking program. Some of the calculations are not sequential in nature. In other words, the operation on certain variables does not depend upon the result of the pre-

vious operation on some other variables. Since moving data among the arrays is faster than doing an arithmetic operation, such as add, subtract, multiply, divide or compare, it is advantageous to line up as many variables as possible, then obtain all their results through a single arithmetic operation. In a simple and straightforward program, only one processor is active at one time. For instance, when the AP control is busily performing a certain operation, the sequential processor and the PIO processor are idle. In the passive tracking program design, effort has been made to divide the total data processing responsibility among all three processors and overlap their activities as much as possible.

The passive tracking program is coded in Goodyear floating point and uses the system supplied floating point routines as well as the floating point mathematic subroutine package developed by BCS. It is desirable to code the program in floating point to preserve the computing accuracy and obtain meaningful results. The entire program including all the necessary library subroutines easily fits into the STARAN-1000 memory without any overlay structures.

A more detailed description of the passive tracking function will be found in Volume II of this report.

1.2.1.2 Performance Measurement (Parallel and Sequential)

The sequential version of passive tracking algorithms was implemented on an IBM 360/65. The parallel version of the passive tracking algorithms was implemented on the GAC STARAN. Identical test scenarios were input to both the serial and parallel programs and the output was compared. Due to the difference between the hardware and software of the IBM 360/65 and GAC STARAN, and due to the difference in programming techniques required for the two implementations, it was difficult to predict the degree of agreement that can be achieved. However, the difference between the output from the serial and parallel programs should not be greater than some tolerance. This tolerance was determined and agreed to be 10% by both BCS and RADC personnel. In an overwhelming number of instances, the variation in track position values after 60 scans of processing had a tolerance of less than 1% - well within the agreed upon tolerance. The test scenario was designed to exercise each section of code in the program. Routines designed to output intermediate values sufficient to checkout each major function of the parallel program were added to the serial program. The output from these routines was used to checkout the parallel program.

As indicated in the previous section, the arrangements and dependencies among the processors required extensive timing and detailed per-

formance measurement. In particular, in measuring the AP processing time (exclusive of PDP-11 I/O operations), it was determined that this processing time was dependent on the speed with which PDP-11 disk operations were performed. Therefore, the various cases and modes of program execution were timed at least 4 times to produce an average value. Figure 2 shows the average values for each mode (self-passive, cooperative-passive, and active tracking) in the parallel version. The crossover point, as measured by the number of tracks processed in 60 scans for equal time intervals is seen to be about 20 tracks for self-passive as well as cooperative-passive modes. In the active case, the parallel version is more than 3 times faster for 64 tracks.

More detailed results of performance measurement may be found in Section 2.4 of Volume II of this report.

1.2.2 Coordinate Conversion Function

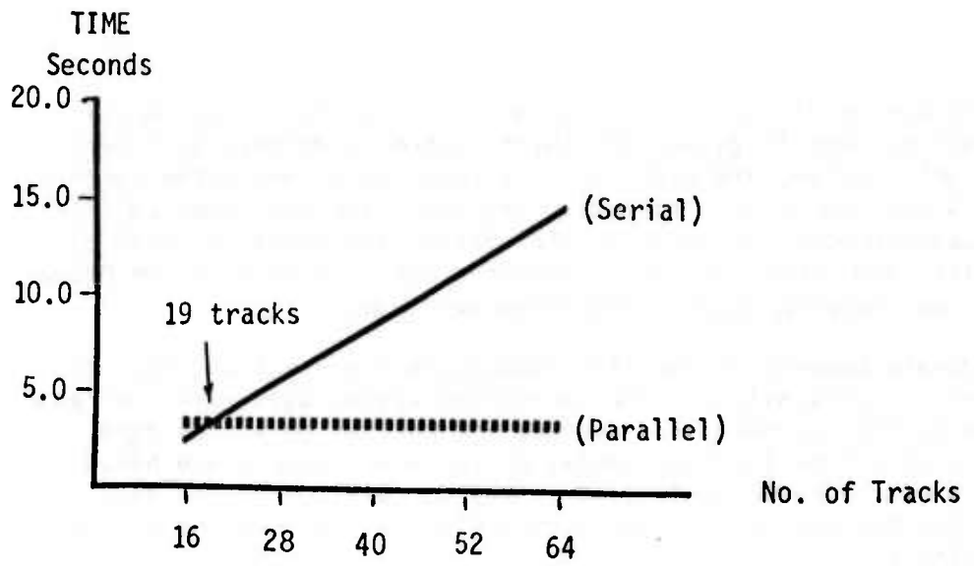
1.2.2.1 Parallel Coordinate Conversion Implementation Description

The function of the Parallel Coordinate Conversion Implementation is to correlate radar data with navigation data, then convert this data to the Command and Control Coordinate System (CCCS) for use by the tracking algorithms.

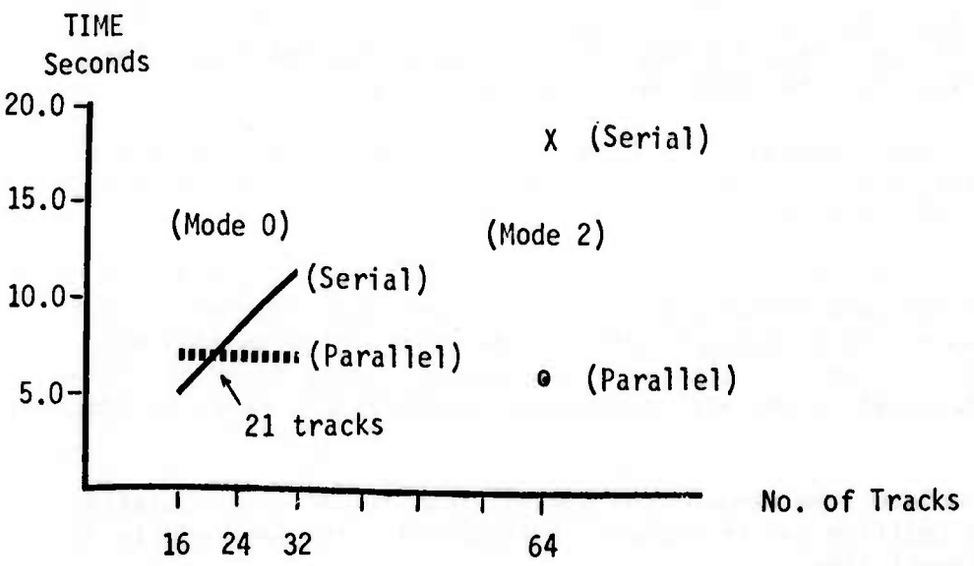
There are four coordinate systems involved in coordinate conversion. One of the two intermediate systems is the Sensor System (SS). It defines a target location in reference to the AWACS. The system origin is the center of the rotodome, with the Y axis directed forward along the longitudinal axis of the rotodome and the X axis directed along the right wing. The X axis is the spin axis of the antenna and is directed positive upward.

The other intermediate coordinate system is the Common Sensor Coordinate System (CSCS). This system is the intermediate system in the conversion of radar target data from the Sensor System (SS) to the Command and Control Coordinate System (CCCS). The CSCS origin is the center of the radar rotodome. The negative Z axis is directed toward the center of the earth. The XY plane is perpendicular to the Z axis with the Y axis directed toward the North Pole and the X axis directed east.

The final system for radar target data is the Command and Control Coordinate System (CCCS). This is the system in which all tracking is done. The CCCS is defined as a two-dimensional coordinate system in a plane tangent to the earth's surface. The point of tangency is the origin of the coordinate system. The principal axes are the U, or east axis and the V, or north axis.



Mode 1 (Self-passive tracking)



Mode 0 (Cooperative passive tracking) and Mode 2 (Active tracking)

FIGURE 2 PASSIVE TRACKING PARALLEL AND SERIAL TIMING COMPARISON

The fourth coordinate system is the Earth System (ES) in which all the navigation data is given. The Earth System is defined as a terrestrial grid system. The position of a point on or above the earth's surface is defined by its latitude, longitude, and altitude. Latitude, λ, is measured positive north of the equator and negative south of the equator, and longitude, Λ, is measured positive east of the Prime Meridian and negative west of the Prime Meridian.

The coordinate conversion parallel implementation has two steps. The first step is correlation of the navigation system data with the radar report data. This is necessary because the navigation system data is updated on a time interval basis, while radar reports are based on azimuth of the radar contact. Once the correlation of the data is complete, the second step can occur which is the coordinate conversion itself.

The correlation of the navigation system data with the radar report data is accomplished as follows.

There are two reports containing navigation systems data, the Azimuth-Time-Attitude (ATA) report, and the Navigation (NAV), or position, report. The ATA report contains the coincident radar rotodome aximuth, the time that rotodome was in that position, and the AWACS attitude at that time. The NAV report contains the AWACS position and the time that the AWACS was in that position.

The radar report contains the radar rotodome azimuth at the time of the contact, a delta azimuth, plus the other radar contact data such as range and elevation.

The radar contact azimuth is found by adding the delta azimuth to the radar rotodome azimuth, both of which come from the radar report. Then given the radar contact azimuth, the time of the contact and the AWACS attitude at the time of the contact can be found by linearly interpolating the ATA reports on the ATA report radar rotodome azimuth.

Finally, knowing the time of the contact from the ATA interpolation, the AWACS position can be found by interpolating the NAV reports on the NAV report time.

After determing the AWACS position at the time of radar contact, the second step of the program performs a series of coordinate transformation to convert the radar report data to the Command and Control Coordinate System.

The coordinate conversion parallel implementation consists of five operations. They are 1) reading data from the disk into bulk memory, 2) taking data from bulk memory and putting it in the arrays, 3) doing the coordinate conversion on the data, 4) taking the converted data from the arrays and placing it in bulk memory, and finally, 5) taking the converted data from bulk memory and placing it on the disk.

There are three programs that accomplish the above five operations: the AP Program, the Coordinate Conversion Program, and the PIO Program. The AP Program is the main program for the coordinate conversion function. It controls the execution of the Coordinate Conversion Program through the setting of certain variables and by when it is called. The AP Program also controls the PIO Program through the use of interlocks and array assignments. Besides controlling the other programs the AP Program reads data from the disk into bulk memory and writes the converted data from bulk memory onto the disk. The program was developed using fixed point arithmetic.

The PIO Program loads and unloads the arrays. Part of the loading operation is to interpolate the ATA reports and to correlate the NAV report data with the radar report data. It takes the data from bulk memory that was read from disk by the AP Program and moves it into the arrays for use by the Coordinate Conversion Program. It also takes the converted data, generated by the Coordinate Conversion Program, from the arrays and places it in bulk memory for writing onto the disk by the AP Program.

The AP Program and Coordinate Conversion Program operate on one set of arrays and one set of input/output buffers, while the PIO Program operates on the other set of arrays and input/output buffers. This concept is shown more completely by Figure 3 Data Flow Control Diagram. For example, while the PIO Program is operating on data set N with output buffer 0 and data set N + 2 with input buffer 0 in arrays 0 and 1, the AP Program is operating on data set N - 1 with output buffer 1 and data set N + 3 with input buffer 1, and the Coordinate Conversion Program is operating in arrays 2 and 3 on data set N + 1.

1.2.2.2 Performance Measurement (Parallel Version)

Two sets of data were used for program validation as well as performance measurement. The STARAN computed results were compared against the hand calculated results with good agreement. Some slight discrepancies might be introduced by the approximation used in the computation of trigonometric functions and/or the loss of significant bits

Radar Report Records read from disk to input buffer (Under AP Program Control)	Input moved from input buffer to AP arrays (Under PIO Program Control)	Coordinate Conversion (CC) (Under Coordinate Conversion Program Control)	CC output moved from AP arrays to output buffer (Under PIO Program Control	Write output buffer to disk (Under AP Program Control)
Data Set N+2 Input Buffer 0	Data Set N+1 AP Arrays 2-3 Input Buffer 1	Data Set N AP Arrays 0-1	Data Set N-1 AP Arrays 2-3 Output Buffer 1	Data Set N-2 Output Buffer 0
Data Set N+3 Input Buffer 1	Data Set N+2 AP Arrays 0-1 Input Buffer 0	Data Set N+1 AP Arrays 2-3	Data Set N AP Arrays 0-1 Output Buffer 0	Data Set N-1 Output Buffer 1
Data Set N+4 Input Buffer 0	Data Set N+3 AP Arrays 2-3 Input Buffer 1	Data Set N+2 AP Arrays 0-1	Data Set N+1 AP Arrays 2-3 Output Buffer 1	Data Set N Output Buffer 0

Time ↓

Data Set Processing →

FIGURE 3 DATA FLOW CONTROL DIAGRAM

in fixed point multiplication and division.

The first set of data was a single radar report with associated ATA and NAV reports. The data was loaded into arrays 0 and 1. The radar report was then replicated to create a set of 64 identical reports withassociated NAV, ATA data in each array. The second set of data included 128 different radar reports (64 reports in array 0 and 64 reports in array 1), NAV reports and ATA reports. Several radar reports were picked for hand calculation through the whole coordinate conversion process. The results agreed with those of the STARAN very well.

The program execution time estimates on STARAN as well as on IBM 360/65 are presented. The actual execution time on STARAN for both data sets are listed. The ATAN subroutine which computes the arctangent of the ratio of two numbers is slow and time consuming. Depending upon the ratio of the input operands, the ATAN routine goes through different paths and results in varying execution times. A conservative value of 8 milliseconds was used for ATAN routine timing estimates. However, the actual timing fluctuates between 4.3 milliseconds and 5.8 milliseconds. This variation has a big impact on the timing of total coordinate conversion program since the ATAN routine consumes a significant portion of the total timing.

The Parallel Coordinate Conversion Program consists of three major subroutines. The timing estimates for the STARAN are derived by counting instructions in each subroutine and then using the timing information in the STARAN manuals.

The timing estimates for the IBM 360/65 are based on the processing of 128 radar reports with corresponding NAV and ATA data. This is equivalent to a single pass of the Parallel Coordinate Conversion Program. The use of floating point arithmetic in Fortran IV is assumed. The timing estimates for the IBM 360/65 are derived by analyzing the operation in each of the Parallel Coordinate Conversion Subroutines. It is then determined that how many sequential operations were needed to perform these functions. The timing of the arithmetic subroutines and the floating point operation is based on the IBM System/360 Fortran IV Library Subprograms and the IBM System/360 Model 65, Functional Characteristics.

FIGURE 4 PERFORMANCE OF COORDINATE CONVERSION PROGRAM

Timing in Milliseconds / Programs	ESTIMATES		ACTUALS	
	IBM 360/65	STARAN	Data Set 1	Data Set 2
Coordinate Conversion Subroutine 1	197.077	25.414	20.018 (4)	20.111 (1)
Coordinate Conversion Subroutine 2	46.398	18.289	13.970 (5)	14.009 (2)
Coordinate Conversion Subroutine 3	85.506	20.797	15.437 (6)	14.756 (3)
TOTAL	325.981	65.500	49.425	48.876

Timing in msec / Note No.	1	2	3	4	5	6
Subroutine ATAN Actual Timing	4.346	5.838	4.980	4.338	5.805	5.791

1.2.3 Radar Data Correlator

1.2.3.1 Description of Radar Data Correlator (RDC)

The AWACS surveillance radar employs two digital computers; the first one is the Digital Data Processor (DDP) which accepts the digitized raw data and performs the functions such as FFT, CFAR, elevation centroid etc. The output of the DDP which consists of range gate number, filter number, amplitude, elevation, PRF number, target number and average filter level (CFAR) of target report(s) is passed onto the Radar Data Correlator as its input. All the input data are then ordered by ambiguous range and by filter number within the same range. The RDC performs range centroid by combining all contiguous filter numbers with the identical range gate numbers. It creates a slant file per each centroid report. All the files are then reordered by centroided range rate. The Chinese remainder theorem is used to resolve range ambiguities and candidate target files are formed. A candidate file is established by a successful correlation in velocity and elevation among three contiguous slant files. The candidate target files are ordered by the range and by the velocity within the same range. Due to the radar beam width, a target may be seen by the radar in 6 or more slants. In other words, 6 or more files may belong to a single target. The function of azimuth extension is to combine the latest target file with previously established candidate targets if they correlate in velocity and elevation. Due to the size or the movement of a target, it may fall in up to three contiguous unambiguous range gate over 6 or more slants.

Those candidate target files which straddle over range gates but display proper azimuth, velocity and elevation relationship are identified and combined into one target file.

1.2.3.2 Parallel Radar Data Correlator Design

The original AWACS RDC does not run in strict real time. Under heavy loading, the processor runs as much as five seconds behind real time. It is allowed the option of estimating the number of targets in on effort to catch up. The presented parallel design aims for maximum processing speed and intends to run 100 milliseconds behind real time independent of the target number.

At the beginning of the processing cycle, the AP collects four slants worth of raw digitized radar data and loads them into arrays. These data are processed in all four arrays in unison. It takes the same

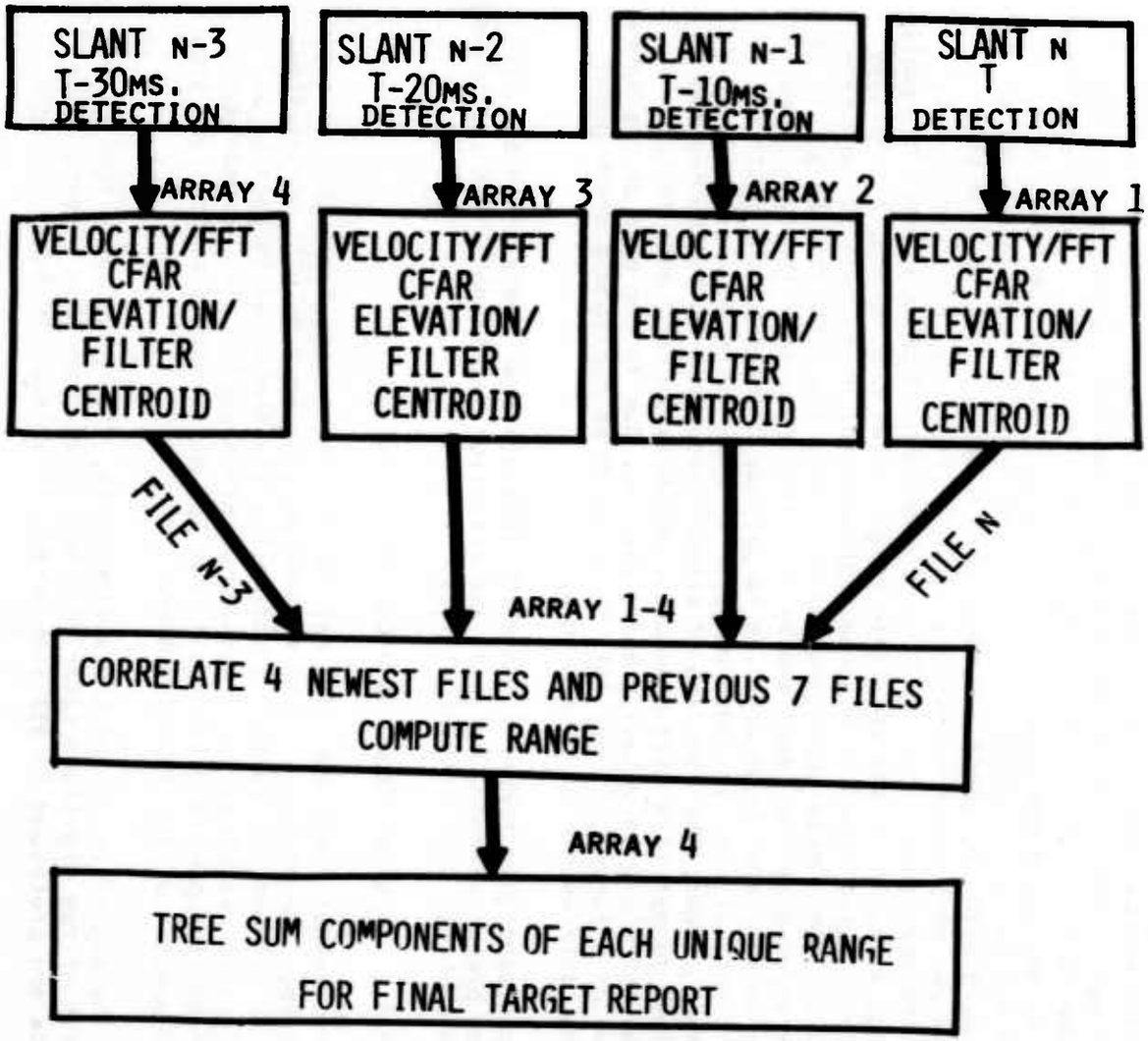


FIGURE 5 PARALLEL RADAR DATA CORRELATOR

amount of time to process one slant worth of data as it does four slants worth. Figure 5 shows the design configuration of the Parallel Radar Data Correlator. On the upper half of the page, the DDP functions such as FFT, CFAR, Elevation Centroid are shown as recommendations since they all are suitable for parallel implementation. They are not, however, within the scope of this design. The Parallel RDC function starts from the Filter Centroid which is performed on data of each slant (or each PRF) in corresponding array respectively. The centroiding compacts the data by a factor of about 5 to 1.

After the filter centroid process, the resultant files, n through n-3 are shuffled among the arrays and are arranged with older files n-10 to n-4 for parallel correlation as shown in Figure 6. Target returns that have similar range rate and elevation are identified. Target files with sufficient number of correlations (such as 3 of 8 or 4 of 8) are flagged for further processing.

The correlated files are spread and re-arranged in all four arrays for the range resolution computation. If the answer of the Chinese Remainder Theorem computation exceeds the radar maximum range test or fails the signal to noise ratio tests, it is deleted. All the surviving files go through a set of parallel arithmetic operation to form the final radar reports. An example of some of the computation involved is shown below. All parameters are properly aligned to minimize the number of arithmetic operations. Each array can handle the arithmetic for up to 16 unique targets in parallel.

A = $\sum$(Signal Strength) F = $\sum$(Filter X Signal Strength) E = $\sum$(Elevation X Signal Strength) Z = $\sum$(Azimuth X Signal Strength) Quality = Number of Observations	6 ADDS

Following the six adds, one divide will be performed.

F ÷ A E ÷ A Z ÷ A	One DIVIDE

The result of performing the above six adds and one divide is a final target report (output) for each target.

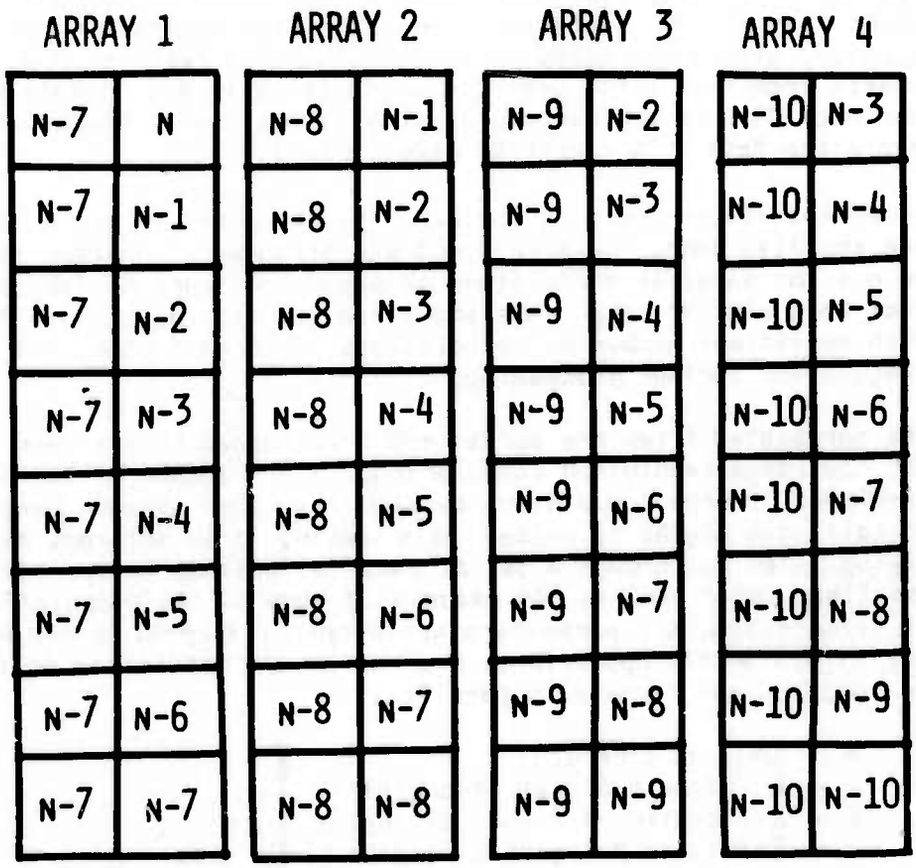


FIGURE 6 ARRAY SETUP FOR CORRELATION

1.3 CONCLUSIONS

A basic conclusion of this study is that the parallel version of the passive tracking algorithms are at least as accurate and are as much as three times faster (based on 64 tracks) than that for the serial version. In fact, the serial version matches the parallel version while operating with about 20 tracks. This is an unexpectedly low crossover point.

It was known in advance that the passive tracking algorithms contain much inherent (first order) parallelism. However, the design for the parallel version went further: it sought diverse sets with which to perform simultaneous operations (2nd order parallelism) and simultaneously used the AP and PIO to great advantage in shuttling data back and forth to support the 2nd order parallelism. This tended to reduce some data movement time to near zero resulting in highly parallel and fast processing. Finally in analyzing the program for timing bottlenecks, it was determined that in a floating point environment on the AP, the Goodyear floating point routines account for as much as 95% of the processing time spent. Thus to minimize the AP processing time requires merely that the number of floating point operations to be minimized.

In contrast, the coordinate conversion program relies solely on fixed point arithmetic. Although a complete timing analysis did not occur, the parallel version shows a performance time advantage over a theoretically designed sequential version.

A study of the coordinate conversion design and timing studies highlighted the conclusion that for much of the time, the PIO is idle. The PIO is responsible for the shorter I/O and data interpolation. The AP is responsible for much lengthier computational operations. The exact amount of PIO idle time was not measured. An additional conclusion is that discrepancies between hand calculator results and STARAN results may be caused by differences in the way each handles trigonometric functions and multiply and divide operations.

The Radar Data Correlator Design has not been implemented but appears to be a feasible endeavor. It appears most desirable to load four scans of raw data at a time into the arrays for processing them in parallel. The design however allows for no concurrent I/O processing. Based on design studies alone, there doesn't appear to be an overwhelming timing advantage with a four array configuration. However, system performance may be improved by a variety of means: increasing the azimuth extension from 6 to 8; improving the efficiency of deghosting; using a 3 out of 8 or 4 out of 8 correlation criteria.

1.4 LESSONS LEARNED

The purpose of this section is to provide a brief description of unexpected experiences with the RADC testbed consisting of the GAC STARAN-1000, the DEC PDP 11-45 and the HIS 6180 (using the MULTICS operating system) while implementing the passive tracking and coordinate conversion parallel programs. Five areas are covered below. These include design, checkout, validation, timing, and operational considerations.

Currently, system supplied languages for the STARAN consist of APPLE assembler language and machine language (microcode) using the GEN instruction. Depending on the objectives of the program being designed and implemented, one may use a mixture of each or purely one or the other. For passive tracking, the main program and principal subroutines were designed (and coded) for the APPLE Assembler level of activity. Subroutines residing in the pages (floating point package, AP data movement) and macro expansions were designed and implemented at the microcode level to conserve time and space (in that order). This appears to minimize checkout time for APPLE code as well as provide efficient underlying microcode. Microcode disadvantages include: keeping track of an undocumented register known as the HOME register when storing masked to array memory, or while performing mask operations; considering the effects of the flip network on associative loads and stores. APPLE disadvantages include: the need for explicit instructions to transform two or more response store registers (or to transform a response store register while modifying data pointer, block-length, field pointer and/or field length registers); the diagnostic silence when, in a loop sequence of instructions the last labeled instruction is one-to-many (e.g. store response store register masked). These pitfalls serve to underline the need for design caution whatever the level of activity.

As a way of containing the impact of some of these problems, as well as shortening the coding and checkout time, symbolic register status maps for microcode were heavily used. These maps provided a code line correspondence with the symbolic variables of the design, such that computational subprocesses could be highlighted. When such code was checked out, one merely traced any discrepancy between the computed value and the symbolic value. Even before code was checked out, manual run throughs served to detect design, code and consistency errors. As a consequence, the design period, for the BCS floating point package planned for 300 hours, was nearly three times longer than the checkout period which required 104 hours. The amount of source microcode involved, exclusive of comments, was 503 lines. The same idea on a grander scale was applied using field status maps for the passive tracking checkout. Note that all coding was previously completed prior to devel-

oping these maps. Figures 7 and 8 show the form of the register status maps and field status maps respectively. Horizontal lines are used to separate subsections of code. Also each page of a map recapitulates the previous page.

During the checkout process, it was seen that heavy reliance on the manuals exclusive of the STARAN itself was a poor policy to follow. For the most part, the manuals were a true reflection of the machine state. However, each of the manuals had a number of glaring omissions whose remedies are described below.

The reference manual should discuss the HOME register, shift and flip networks in much greater detail. The APPLE and MAPPLE manuals should be modified to reflect the number and content of lines of code for each "one-to-many" instruction. The SHIFTY and ROTATY instructions, when shifting or rotating between arrays, should use only AP macros and subroutines not PIO routines, as is presently the case.

Timing data should be provided for each instruction described. The User's Guide, although it provides a valid description of SDM commands, does neither offer legal orderings of its commands nor describe its current overlay structure (e.g. inspect and change commands intermixed with performance monitor or external function commands result in wasted runs). This means that the user must use trial and error to determine what works.

As a further checkout aid, careful configuration control was instituted at the onset of program checkout. In the passive tracking program, all modifications to the program were dated (in line image, column 73-80) and deletions and replacements were commented to provide checkout continuity and flexibility. At the end of checkout, nearly 80% of the code had undergone revision.

A necessary ingredient to any checkout effort is the means to crosscheck program computations against hand calculations, published tabular values (if they exist), or other tabular computer calculations. The passive tracking program used the second and third types of crosschecks. The Coordinate Conversion program was only able to use the first type of crosscheck. While validating the floating point trigonometric routines, tabular values for cosine and arctangent were converted (via APL on the 370/168) to hexadecimal format and compared with hexadecimal dumped output from the STARAN array. The Passive Tracking output (track position) of the serial version (already decimalized) was compared with a decimal converted dumped output (via PL/I thru MULTICS on the HIS 6180) from the STARAN array. Such con-

FIGURE 7 (AP) REGISTER STATUS MAP

CODE LINE	AS	CR	BL	DP	FL1	FL2	FP1	FP2	FP3	FPE	X	Y	M	ARRAY

FIGURE 8 FIELD STATUS MAP

PAGE, ROUTINE, CODE LINE	FIELD 0				FIELD 1				FIELD 2				FIELD 3				FIELD 4				FIELD 6			
	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3

FIELD STATUS MAP (Continued)

PAGE, ROUTINE, CODE LINE	FIELD 7				X				Y				M				ASH
	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3	A0	A1	A2	A3	

FIELD STATUS MAP (Continued)

PAGE, ROUTINE, CODE LINE	INTERLOCKS				ARRAY ASSIGNMENTS					AP REGISTERS					
	IL2	IL3	IL4	IL5	AP	PIO	ASH	CR	FL1	FL2	FP1	FP2	FP3		

FIELD STATUS MAP (Continued)

PAGE, ROUTINE, CODE LINE	FPE	BL	DP	PC	FL1	FP1	PIO REGISTERS FP2	FP3	BL	DP	BCW1	BCW2	BCW3

FIELD STATUS MAP (Continued)

PAGE, ROUTINE, CODE LINE	HSDB STORE	BULK CORE STORE

version aids are essential to program checkout and it is hoped that some of these conversion programs developed by RADC personnel become permanent additions to the STARAN system.

During the timing of the Passive Tracking program, several observations were made. PIO routines cannot be timed directly when they are (inter)dependent upon AP processing. Coupled with this problem is a lack of timing data for PIO instructions. One can only surmise that the instruction time falls between that of corresponding AP instructions when loaded into the high speed data buffer or bulk core memory (timing estimates for Passive Tracking assume a maximum time corresponding to bulk core instruction loads). On the AP side, it appears that Goodyear floating point instruction calls (in a pure AP program) account for up to 95% of the total AP time measurement. This is extremely useful in estimating program times and indicates that the most fruitful areas for program optimization lie in minimizing the number of floating point operations performed.

During the checkout process, STARAN communication was through three related modes of operation: While in Seattle, a remote terminal was used to connect with the MULTICS system in Rome. Runs could be submitted on a batch basis via the STARAN "SysDaemon" passing it through to the PDP/11 and finally to the STARAN. While onsite at RADC, either Remote Terminal usage or standalone usage was possible. In the latter case, only the PDP/11 and the STARAN were accessed.

It turns out that remote access from Seattle (or anywhere other than RADC) is the least desirable alternate. Advantages include: No travel funds required and ability to make somewhat efficient use of programmer activities. Disadvantages include unreliable communication hookup and line quality which is wildly erratic, also unscheduled down time among each of the accessed computers as well as the small window during which communication occurs all serve to inhibit large scale activities. At RADC, the communication problem is minimized and standalone is feasible when MULTICS is down.

To exploit these various methods of access, all procedures on or via MULTICS were designed for minimum computer response online and maximum computer response offline (via high speed printer). Second, all necessary source and object files were duplicated on the MULTICS and PDP-11 systems to be prepared for MULTICS (un)scheduled down time. Of course, less could be done if either the PDP-11 or STARAN went down. By the end of the study period (March 1976), it appeared that the MULTICS window was narrowing due to the daily peak loading of MULTICS from 0900-1200, and 1300-1700. Such peak loading leads to system pre-

emption, extremely slow response times and time degradation (particularly time limits) of any STARAN SysDaemon jobs containing internal MULTICS interfaces (e.g. SDM post processing of array and memory dumps onto MULTICS files).

It is not the intent of this section to belabor the tribulations of a project and computer systems that are involved. Rather, it is hoped that some of these considerations and experiences may be heeded by others - not merely the project participants on future projects of a similar nature.

1.5 RECOMMENDATIONS

Due to the learning process in designing, and implementing this Passive Tracking parallel program, some recommendations are in order:

(1) The program should be redesigned to optimize the AP/PIO interactions such that both total program time and AP idle time are reduced. Considered should be data movement routines that operate efficiently on the AP side as well as on the PIO side.

(2) Portions of the program were written in pure microcode while other portions were heavily Apple assembler language oriented. Certainly the microcode, once checked out, is cleaner, faster, more compact code. However, if minimizing program checkout time is the main objective, then assembler language should be utilized throughout. On the other hand, if minimizing program storage and timing is the objective, microcoding of key subroutines and processes should be exploited from the onset. It is recommended that current (MACRO and) APPLE Assembly routines be scrutinized for improvements using microcode.

(3) An extremely useful adjunct, while running the passive tracking program was the development by RADC of floating point conversion routines. Two routines were used, one to convert unformatted binary to decimal values and the other to convert formatted (hexadecimal) ASCII to decimal values. The main feature of these routines was the speed at which many thousands of numbers could be converted. It is recommended that these routines be made available within Goodyear system software as an addition to the post processing capabilities of SDM.

For Coordinate Conversion, it is recommended that the program be optimized to reduce that PIO idle time which contributes to an overall time reduction. As one possibility, a more dynamic array assignment algorithm should be added to better balance the PIO and AP processors.

For Radar Data Correlation, it is recommended that the program design be implemented to test and determine the amount of real-time performance enhancement that is possible.

*U.S. GOVERNMENT PRINTING OFFICE: 1976-714-025/34